LSST DESC Framework Use Cases

Jim Kowalkowski, Marc Paterno, Saba Sehrish, and Scott Dodelson

Revision 11

Contents

1 Introduction

1.1 Vision

There are three key features about LSST analysis that are relevant to building a framework:

- Hundreds, if not thousands, of scientists will want to analyze the data
- On most projects, the work will *not* be done by a single person so that the code that enables them will be written by many people
- There are many tools available to the scientists, and these will grow as the people in both the project and the science collaboration produce more software

A framework (e.g., see Fig. 1) that accounts for these features must allow for sharing, so it will need to include repositories in which sets of codes are housed. Since projects have multiple steps, data formats must be well-defined. One user needs to know how to input data from modules written by another user and how to pass outputs so other users can benefit. A final important part of sharing code is the flexibility that allows different users to code in different languages.

The framework also needs to make analysis simple; that is, to allow the scientists to spend the bulk of their time developing code for their analysis, not installing modules and determining what version of program Y is needed to run module X. Particle physics groups have learned to deal with packaging (and indeed, the LSST DM stack uses `eups`), so packaging will be an essential component of the framework. Instead of `make` files, users will run code by specifying which modules to include in a `config` file. Ideally, users will not even have to worry about where their code is running; the framework will determine the needed resources and send it to an appropriate machine. Finally, a coherent framework will be most helpful with a single unified support team.

1.2 Purpose of this document

The purpose of this document is to start the process of implementing this vision by determining the requirements[1] for the LSST Dark Energy Science Collaboration[2] Level 3 Software Framework[3], This is best done by cataloging a set of *use cases* that will determine these requirements and therefore determine the nature and scope of the framework.

The authors' task in assembling this catalog has been to discuss some scientific goals with the several LSST DESC Working Groups, to sift through the result of the discussions to identify use cases for the framework and the development and execution environment, and to identify some of the existing software tools that would be used within the framework. This catalog will aid in the identification of requirements for the framework by helping to

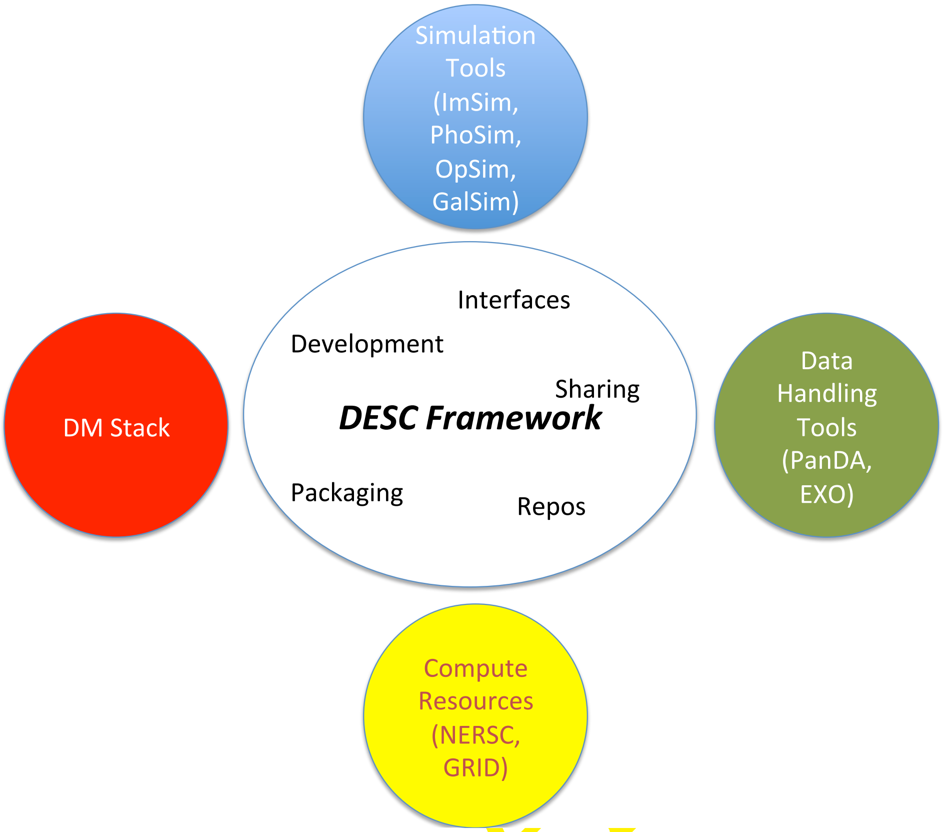
Simulation Tools (ImSim, PhoSim, OpSim, GalSim)
Interfaces
Development
Sharing
DESC Framework
Packaging
Repos
DM Stack
Data Handling Tools (PanDA, EXO)
Compute Resources (NERSC, GRID)

Figure 1: A Level-3 software framework will be driven by scientific uses and include an environment that enables hundreds of scientists to collaborate. It will enable the scientists to easily make use of simulation and Data Management tools from the project; data handling tools that have been developed for other projects; and access large-scale computing resources.

assure that the user community's needs, as described in the use cases, are the needs that drive the requirements gathering process.

1.3 Overview

A *use case* is a list of steps that will be taken in a given analysis. The purpose of writing use cases is to aid in gathering functional requirements for a system.

The steps in use cases often involve mention of the use of software components (either preexisting or hypothetical) that are treated as "black boxes" by the use case; their internal details are not described. In the context of the framework, we call these components *functional elements*, and they are best viewed as atomic units. We describe their inputs and outputs, and their purpose, but do not describe how they do their job.

Functional elements can be executed in isolation from other functional elements. The most trivial use of the framework is to execute a single functional element. This is at least potentially different from running a functional element in isolation because running in the context of the system implies running in the environment defined by the system, and taking inputs and producing outputs in one of the formats defined by the system and using the framework facilities for finding inputs. In general, even a single-element "workflow" in the framework will make use of framework facilities.

The remainder of this document is organized into two major sections. In part 2 we introduce some of the functional elements which form the "building blocks" from which tasks using the L3 software framework will be built. A *functional element*, in this document, is a program or subprogram that a scientist will use to complete one part of a task described in a use case. These functional elements will be run as part of a DESC framework program, and thus are atomic parts of the system described in this use case catalog.

Part 3 is the main body of the document, which consists of a catalog of use cases for the framework.

2 Catalog of functional elements

2.1 Weak Lensing Functional Elements

2.1.1 Sheared-image reformatting tool

Description

Take a sequence of images of galaxies with little or no shearing, apply a known shear to each, and produce an input catalog for PhoSim.

Input types

1. A sequence of FITS files containing:
    - postage stamp images, and
    - a catalog containing the position of each galaxy.
2. Either one shear to be applied to every galaxy OR a catalog of shears containing one for each galaxy.

The images should be at a resolution better than LSST will obtain, and have a PSF much smaller than LSST's PSF. Each postage stamp image should be of a galaxy that is not strongly sheared.

How inputs are obtained

Images are obtained from an external source, *e.g.* HST. All the files are expected to be local files.

Output types

A PhoSim input catalog containing a true sky image, which includes the applied shear, for each galaxy.

Common follow-up uses

The output is put into PhoSim to create an image that includes the effects of the atmosphere and telescope.

Estimates of time and size

The typical single image is $100 - 1000$ pixels. Typical uses require 10^4 images. The CPU time per image processed is $\ll 1$ second.

2.1.2 Ian's galaxy shearing tool

Description

Take a sequence of images of galaxies with little or no shearing, and description of a foreground galaxy cluster, and produce an input catalog for PhoSim.

Input types

1. A sequence of FITS files containing background galaxies:
   - postage stamp images, and
   - a catalog containing the position, redshift and colors for each galaxy in these postage stamp images.
2. A description of the 2-dimensional (or optionally 3-dimensional) mass density along the line of sight to each background galaxy.
3. A FITS file containing a foreground galaxy cluster, with a postage stamp image for each galaxy in the cluster, including colors.

The postage stamp images should be at a resolution better than LSST will obtain. Each image should be of a galaxy that is not strongly sheared.

Their are several possible representations of the mass density. A 2-dimensional representation would require a few million pixels. The third dimension of a 3-dimensional representation would have much less resolution than the other two dimensions.

QUESTION: *We want to enumerate what input types the existing tool can read.*

How inputs are obtained

Input images are typically in FITS files, obtained *e.g.* from HST.

The description of galaxy clusters may come from an N-body simulation, or from a NFW profile, or from observed clusters.

Output types

A PhoSim input catalog, containing a true sky image, which includes the applied shear, for each galaxy.

Common follow-up uses

The output is put into PhoSim to create an image that includes the effects of the atmosphere and telescope.

Estimates of time and size

The typical single background galaxy image is $100 - 1000$ pixels, in each of 5 optical bands. Typical use would require 10^3 background galaxies. Processing time per image is < 1 second.

The typical galaxy cluster image is a few million pixels; the mass distribution would also contain a few million pixels.

2.1.3 Galaxy shape measurement tool

Description

Measure galaxy shape parameters.

There are many different algorithms, with a preference for forward modeling, using a parameterized model and process it through the effects of PSF and noise, comparing with the data and forming a likelihood in the N-dimensional space.

Input types

A sequence of FITS files containing:

1. Postage stamp images of the galaxies to be measured; these images may be coadded,
2. A set of catalogs, with one entry per postage stamp image. For each image, we need:
   - a mask, possibly one mask for each color channel,
   - a specification of noise mask (noise from readout and photon statistics),
   - a specification of WCS,
   - a PSF model with a covariance matrix for that location.

Each postage stamp image should contain a single galaxy.

If coadded images are used, then there needs to be some processing to get the effective PSF, the (correlated) noise profile, the net WCS, etc.

It is important to provide all of the single-epoch versions of these things so an algorithm that works on coadded images can combine them as needed.

How inputs are obtained

Output types

Outputs may appear in multiple forms.

One possible output is, for each galaxy, an estimate of the shape parameters and the covariance matrix of those estimates.

A second possible output is a set of values of the full N-dimensional likelihood, as a function of the model parameters, where the galaxy shape is described by the N parameters.

Output is generally written as FITS catalogs.

Common follow-up uses

Estimates of time and size

2.1.4 Determine the PSF at the locations of stars in an image

Description

There are different algorithms for extracting the shape parameters of the stars (*i.e.*, the PSF at the location of the star). The first step will be ingesting the images. Second will be applying the algorithm to the image(s); this often involves computing and/or maximizing a multi-dimensional likelihood function. The final step will be outputting a series of shape estimates including errors.

Input types

A FITS file containing:

1. an image with stars, and
2. a catalog of locations of the stars in the image.

How inputs are obtained

The image may be simulated (*e.g.* by PhoSim) or from the LSST data.

Output types

Several possible outputs are identified.

1. A shape estimate (*i.e.*, a set of fit parameters plus covariance matrix for the fit parameters) for each star.
2. Samples of the N-dimensional likelihood (where N is the number of parameters used to describe the PSF) for each star.

Common follow-up uses

Estimates of time and size

Typically a vector of $50 - 100$ numbers are used to describe the shape.

2.1.5 Simulate images of galaxies and stars with GalSim

Description

Some DESC tasks do not require the full LSST instrument/atmosphere realisation provided by PhoSim—for these we can use GalSim to simulate fields containing stars and galaxies.

Simulate images containing stars and/or galaxies, for use in various DESC tasks. This image may or may not contain a PSF, depending on the purpose of the simulation.

GalSim will be used by many DESC tasks to produce images of the sky with a given PSF where the full atmospheric and instrument simulation is not required. It will also be used to produce images of the sky without PSF effects, which can then be used as input to PhoSim, in which case this task is the precursor to task (1) on this list. The usual way to run Galsim will be to write a YAML configuration file with the specifications about what objects to build, noise properties, image size, *etc*. See the Galsim wiki for details.

Input types

1. A list of objects to be simulated, one of:
   - input postage stamp images from the GalSim suite of HST galaxies, or
   - input objects from a model.
2. If a PSF is to be used in the image, this must also be specified.

Model formats for galaxies include

1. de Vaucouleurs
2. Sérsic
3. exponential
4. gaussian
5. arbitrary user-provided image
6. shapelet decomposition

Model formats for stars include

1. Moffat
2. Airy
3. Kolmogorov
4. OpticalPSF (with coma, astigmatism, *etc*.)

How inputs are obtained

Output types

If the purpose of the simulation is to produce an image fits file that can be run through PhoSim, then the output fits file must be included in a PhoSim instance catalogue (which also specifies the observing conditions, *etc*). Otherwise, the image may be analysed by the DM stack, or other algorithms.

Common follow-up uses

Estimates of time and size

2.1.6 Star-Galaxy separation tool

Description

Provide a set of probabilities for each object, or a binary classification of whether an object is a star or galaxy.

Input types

1. A set of images, or coadded images.
2. Catalog of identified objects in each algorithm. Some algorithms use only this information, and do not require the images.
3. Catalog(s) of stars from other survey(s).
4. Template(s) from another survey(s).

Many algorithms require multi-color information.

QUESTION: *What is a template as used in this context?*

How inputs are obtained

Output types

Two different types of outputs are identified, based on the capability of the algorithm being used:

1. For each input object, the probability that the object is a galaxy.
2. A binary classification as star or galaxy.

The algorithm itself is not the point of the task There are many algorithms developed to perform this task. This task will enable any of these to be implemented on the input data.

Common follow-up uses

Estimates of time and size

2.1.7 PSF interpolation tool for a single exposure

Description

From the measured shapes of stars in an image, construct a “PSF field” that can be evaluated anywhere. The field will be evaluated at the locations of galaxies to extract the intrinsic shapes of the galaxies, accounting for the effects of the PSF.

This tool requires input information from a single exposure.

Input types

1. A catalog of shape estimates for stars.
2. Optionally, a series of FITS files containing images of the stars.

How inputs are obtained

The input PSF models are created by a tool like that described in 2.1.4.

Output types

1. The PSF field, along with uncertainties of the PSF field, that describes the PSF at every point in the sky. The PSF field in a given field is typically represented by coefficients of basis functions (*e.g.*, polynomials), and the covariance matrix for the parameters.

QUESTION: *Different algorithms tend to have unique output formats. We would like to enumerate the important formats.*

Common follow-up uses

Estimates of time and size

2.1.8 PSF interpolation tool using multiple exposures

Description

From the measured shapes of stars in multiple images, construct a "PSF field" that can be evaluated anywhere. The field will be evaluated at the locations of galaxies to extract the intrinsic shapes of the galaxies, accounting for the effects of the PSF.

This algorithm tries to estimate the repeating components of the PSF pattern using images from many exposures taken with the same telescope, but with different realization of the atmosphere and optical parameters (such as the defocus value). The process may be iterative (updating the model one exposure at a time), or it may require all the information to be available at once. This obviously has implications regarding the memory requirements for the algorithm. We hope to cast all such algorithms into the iterative mode to ease the resource demands from this task.

Input types

1. A series of FITS files containing multiple images of stars observed in many exposures.
2. A catalog of shape estimates for those stars, for each image.
3. Ancillary data about the exposures, such as Hour Angle, Declination, airmass, *etc.*, which might help identify exposures with similar observing conditions.

How inputs are obtained

Output types

1. The PSF field, along with uncertainties of the PSF field, that describes the PSF for each exposure. The PSF field in a given field is typically represented by coefficients of basis functions (*e.g.*, polynomials), and the covariance matrix for the parameters.

Common follow-up uses

Estimates of time and size

3 Catalog of use cases

3.1 Large Scale Structure Use Cases

3.1.1 Correct galaxy number density for stellar contamination

Immediate scientific goal

Determine corrections for how much the systematic of galaxy number density correlated with stellar density affects/pollutes the galaxy statistics measurements. One possibility is that stars make the sky background different so make galaxy detection more difficult.

The way to correct for this is an active area of research. This use case includes the ability to develop and test different algorithms and identify an optimal algorithm.

One current algorithm is to run an MCMC on the data assuming a polynomial fit for the galaxy and stellar density. This algorithm may need to be run on many different galaxy types and levels of stellar brightness.

Preconditions

Input catalogs of stars and galaxies from simulation, as well as the truth catalogs from which the simulations were drawn, are needed as input.

The size of the catalogs would be of order a thousand square degrees times 50×3600 per square degree, or up to 100 million objects, although not all fields need to analyzed simultaneously. The total area might be divided up into sub-patches $\sim 10 - 100$ square degrees. Each object needs to have stored ~ 10 characteristics. The correlation function is much smaller, trivial. Computing the correlation function is typically an N^2 process. It would be difficult to store all this data or carry out the correlation function calculations on a laptop, although algorithm development could take place on a laptop. To carry out the full analysis, a medium size cluster is needed. The MCMC method roughly fits 10 parameters, so requires many samples.

Steps

1. Determine the star-galaxy correlation function, based on the input catalogs. The correlation function is output in the form of a function of angular separation.
2. Determine the one-point statistics of stars and galaxies (“counts in cells”).
3. Using the star-galaxy correlation function and/or the one-point statistics, apply an algorithm that corrects for this systematic. This could be in the form of an expected galaxy number density for each pixel, accounting for stellar contamination, or a mask that takes a value in each pixel between 0 and 1.
4. Using the mask and/or the expected galaxy density, compute the star-galaxy correlation function for the corrected catalog.
5. Evaluate the residual cross-correlation in the corrected image.

6. Based on the residual cross-correlation, output an error estimate due to this systematic.
7. If the above steps are repeated multiple times, output “scores” for each algorithm (e.g., the residual cross-correlation or the estimated error) to enable selection of an optimal algorithm.
8. Output a corrected number density of galaxies.

Postconditions

The cross-correlation after correction should be small.

3.1.2 Correct galaxy number density for depth variation and track uncertainties

Immediate scientific goal

There are many effects that lead to depth depending on position on the sky: dithering, tiling, seeing, galactic extinction, atmosphere, time varying in detector response. If these are not accounted for, the observed galaxy density will (incorrectly) be reported as tracing these systematics. The “mask” quantifies how many galaxies there should be in a given pixel if these systematics were absent. Different catalogs (corresponding to different choices of galaxies/qsos) will have different masks. So the survey will produce multiple masks. The essence of this task is to transform the different LSST observations into a mask that encodes how deep a given region was observed. The “mask” in principle includes all effects; this particular task does not include the contribution of stellar fields to the mask.

Preconditions

Simulated images from OpSim that were generated given the telescope pointing plus the conditions under which the data were taken. An image must be made of the OpSim output. This could be done by running ImSim or by using healpix to sub-divide pointings. Healpix can generate skies at different resolutions; currently the LSS WG is running over 200,000 pixels on the sky, with ~ 10 elements in each pixel.

Healpix and OpSim run on full sky, while ImSim runs are typically much smaller fields. ImSim generally is computationally prohibitive to generate large areas used for LSS tasks, but it will be used to examine small scale tests.

Each pixel will be of order an arcsecond, and scientists might typically apply this to $100 - 1000$ square degrees, so the total data set is small. Full skies run on a laptop in 10 minutes. For the input to this task, running OpSim itself takes 3 days.

Steps

1. Run OpSim to generate a series of meta-data at given times that include pointing location, filter, predicted magnitude limit etc.
2. For small scale work, run ImSim to turn the OpSim output into a simulated co-added 10-year image and then run the LSST DM tools to generate a catalog of stars and galaxies OR

3. For large scale work, use Healpix, source count laws, and a Milky Way model to generate a map of simulated star and galaxy counts directly from the output of OpSim
4. Using the information generated in the first 3 steps, estimate the mask at each pixel via the ratio $N_{\text{input}}/N_{\text{found}}$.
5. Correct the galaxy density using the mask
6. Cross-correlate the resulting galaxy density with observables such as the stellar density, the mask itself, or other sources with which it should not be correlated.
7. Using these cross-correlation functions, improve the estimate of the mask and then repeat Steps 5 and 6.
8. Evaluate the residual cross-correlation in the corrected image.
9. Based on the residual cross-correlation, obtain an error estimate on the mask.
10. Output a mask and its uncertainty at all pixels.
11. Other outputs include: a corrected map of the galaxy density; an angular correlation function of the galaxies, both corrected and un-corrected; and the range of scales that is unpolluted by uncertainties in the mask.

3.1.3 Inject fake sources into data and rerun DM

Immediate Scientific Goal

Real data (*e.g.* from SDSS) added to fake sources to see if the DM recovers the sources in the presence of realistic fields. The initial goal is to obtain completeness (probability that a source is recovered) as a function of magnitude and size of the object. This task does not require the shape measurement parts of the DM, but rather the detection tools. In principle, the task is to test many elements of the DM: galaxy/star separation, photometric redshift, and magnitude estimation.

Preconditions

A typical field might be 10 square degrees, corresponding to 3 Gigapixels, with two bytes per pixel. Data would be ingested as a FITS file. One component of input is a prior data set, its images. The other is the characteristics of the fake sources. The assumption is that DM has information about the prior surveys. This task relies heavily on running the DM stack.

Steps

1. Ingest fake sources and their characteristics
2. Ingest the "real data"
3. Combine into a single data set
4. Run DM to obtain catalogs with star/galaxy probabilities, magnitudes, and sizes
5. Compare DM characterization of fake sources with known values

Postconditions

Feedback to DM about its current performance and suggestions for how to improve. This will also be used to help informing simulations to make them more realistic.

3.2 Photo-z Use Cases

3.2.1 Train and test photometric redshift algorithms with training and validation redshift data sets, both simulated and real

Immediate Scientific Goal

Apply neural networks and other algorithms on a training set to derive photo-zs and then verify that solution on the remaining (validation) data. The training and validation sets could be separated in several ways. Different people with different codes will operate on similar data sets. Pre-LSST operations, we will be dealing with simulated data and existing data from external sources. For algorithm development, the data does not have to be exactly in the same bands.

Preconditions

1. Redshift and uncertainty and quality
2. fluxes in LSST bands with errors for those same objects

Algorithms parallelize trivially, so the CPU time needed for 10^4 up to 10^5 objects with spectra (and magnitudes in LSST filters) is minimal.

Steps

1. Input spectroscopic redshifts for both training and validation sets
2. Inputs fluxes in LSST bands
3. Training algorithm by running on training set data
4. Run algorithm fine-tuned in Step 3 on Validation data set
5. Compare photometric redshifts from Step 4 against (true) spectroscopic redshifts
6. Determine how well algorithm did
7. Out set of weights for the algorithm(s)

Postconditions

The set of weights for the algorithms is only a few thousand numbers. In DES, it's stored in a flat file that is read in. The results of this training and validation will be used to inform which algorithms will be run and what their weights are. The outputs from this use case will also be used to interpret photo-z measurements.

3.2.2 Derive N(z) for photometric sample using angular cross correlations against spectroscopic redshift data sets

Immediate Scientific Goal

The goal is to use the fact that galaxy are physically correlated to improve estimates of the redshift distribution of the galaxies.

Preconditions

There are two data sets here: 5×10^9 photometric galaxies from LSST (subsets of these) and 2×10^7 galaxies for which we have spectroscopic redshifts.

Positions of photometric galaxies (*e.g.* magnitude, p(z,type), color, *etc* to define subset) and redshift of spectroscopic galaxies and information to make random catalogs. These positions come from data management and external data sets (*e.g.* DESI). Another input is weak lensing or CMB maps because those carry information about the bias, and $p(z)$ is degenerate with the evolution of the bias (db/dz).

There is a wide range of time scales for different codes, and the collaboration will likely try out different codes. Tree codes for 2-point functions are fastest; on large scales, can pixelize and run in several hours. Fourier-space estimator might be most efficient/fastest. Rough estimate per correlation function ranges from an hour to a day on a single processor. There are parallelized codes. Output sizes are trivial.

Steps

1. Ingest photometric and spectroscopic galaxy information
2. Measure correlation functions: auto- (photo-photo and spectro-spectro) $w(\theta, specz)$ and cross-correlation (photo-spectro) functions (2-point) for subsets of these.
3. Obtain error estimates on these 2-point functions.
4. Combine prior estimates of $p(z)$ with correlation functions to obtain improved estimate of $p(z)$
5. Propagate each galaxy's improved $p(z)$ to form $N(z)$, an estimate of the full redshift distribution
6. Obtain an error estimate for $N(z)$

Postconditions

$N(z)$ for each subset, a handful of numbers, used in many cosmological analyses. For example, weak lensing analyses require $N(z)$ for several different subsets, typically of order 5. This method can be used to refine the photo-z algorithms.

3.3 Strong Lensing Use Cases

3.3.1 Estimate external convergence along the line of sight to a lens

Immediate Scientific Goal

Quantify systematic errors on strong lensing due to weak lensing by external mass distributions. Improve algorithms for modeling these mass distributions and reducing systematics by looking at photometric galaxies along the line of sight. Goal is to not be dependent on simulations but rather to generate an algorithm that reconstructs the mass distribution in cone from observed galaxies and observed weak lensing field. Need simulations to test algorithms, to show that we can infer mass distribution accurately.

Preconditions

Well-measured shears, mock photometric catalog with stellar mass estimates and photo-z estimates, true κ_{ext}.

Might not be able to consider environment separate from lens; in that case would need to extend algorithm to include lens as well

Small ($5'$) galaxy catalog (RA,DEC,photo-z,luminosity, stellar mass, shapes) in a single light cone; next level up is collection of 10^4 of these, together called a training set.

Millennium Simulation lightcones (in hand). Bologna Lens Factory lensed quasar field lightcones (developed by Euclid SLSWG in Bologna).

Density maps: 2 simulations × 3Mb × 50 slices × 10,000 lightcones = 3Tb.

Galaxy catalogs: typically 1/3 of image size = 1Tb.

Steps

1. Obtain simulation products
   - True density maps in redshift slices, to train inference code
   - Mock photometric galaxy catalogs (including shapes), for input into inference code
   - True environment parameters (κ_{ext}, and γ_{ext}, or ring+QSO images), to test recovery currently one single nuisance parameter per lens
2. Implement mass model within Pangloss[1] software write function that inputs galaxies in redshift slice (ra,dec,mass) and output plausible mass distribution
3. Train Pangloss software on thousands of calibration lightcones infer the hyper-parameters (e.g., stellar mass/halo mass relation, c-M relation, recipes for guessing group shapes are) that are used by the software; the sims will set up priors on these hyper-parameters
4. Run Pangloss on test lightcones, possibly run MCMC over 10-dim space, record PDFs for κ_{ext}
5. Combine PDFs and probe biases and precision

Postconditions

Mass distribution along the light cone (intermediate) → PDF for external convergence.

3.4 Weak Lensing Use Cases

3.4.1 Compare the effectiveness of PSF interpolation schemes

Immediate scientific goal

Quantify the difference between the true PSF field and estimated PSF field, and measure statistical properties of these differences, such as the two-point functions. Alternatively,

[1] Pangloss is a homegrown python program, developed over 3 years (*e.g.* a student research project), which will be used many times in this use case, perhaps hierarchically (on small scales and large scales).

the difference between the statistics of the galaxy shape field inferred from the true PSF and that inferred using the estimated PSF can be used as a metric.

Preconditions

Instance catalogs for PhoSim including field of stars and telescope condition.

Some tests use previous statistical information, so input might include results from previous runs.

Steps

QUESTION: *Someone should check all these*

1. Ingest positions of stars.
2. Run PhoSim with stars at their actual positions.
3. Record observed positions and estimate the PSF at those positions using the shapes of the stars
4. Interpolate to obtain an estimate of the PSF at a new (disjoint) set of positions.
5. Run PhoSim a second time at the same new positions to obtain estimates of the "true" PSF.
6. Measure the statistics – correlation function and power spectrum – of the PSF obtained both ways

.

Postconditions

A set of power spectra or correlation functions, or other quantitative measurements of how well the interpolation worked.

3.4.2 Compare output measured galaxy properties to input truth values

Immediate scientific goal

This is a validation/calibration task: determine how well our algorithms are working.

Compare the input catalogs given to PhoSim to the output of various tasks (*e.g.* galaxy shapes).

The quantitative comparisons of truth and output will vary depending on the cases. Sometimes, they will be statistical as in use case 3.4.1. Other times, we can look at individual objects on a case by case basis and require agreement to very high accuracy.

Preconditions

Need the "true" properties of the input galaxies; this might include more information than is contained in the PhoSim instance catalogs.

Steps

1. Ingest true shapes of input galaxies

2. Run ImSim to generate realistic images
3. Separate stars from galaxies
4. Measure the shapes of the stars
5. Interpolate the stellar PSF's to get the PSF across the fields
6. Use this PSF field together with measured galaxy shapes to estimate ellipticities
7. Compare shape estimates from above with true input shapes

Postconditions

Quantitative measurements varying with the application.

Comments

An important related, overlapping task is validating PhoSim. Determining where the responsibility for any differences lie is an important consideration.

4 Glossary

Postage stamp: An image that contains only one object.

PSF: Point spread function. A PSF describes the response of an imaging system to a point-like source (*i.e.* a star). The implementation in software of a PSF can be done in several ways.

The "exact" or "true" PSF means the actual PSF that an infinite S/N point source would have when observed on the focal plane. An "measured" PSF would be the PSF inferred from a particular measurement of a single star. An "interpolated" PSF is the inferred PSF at some other location than where there is an observed star (*e.g.* the location of a galaxy).

QUESTION: *Enumerate the relevant ways of implementing PSFs here. Use case text included "exact" and "interpolated". What does "exact" mean?*

5 Common data types

5.1 Galaxy catalog

A galaxy catalog is logically a "table". The table contains one row per galaxy. Different catalogs may contain different information for each galaxies, as columns.

The data include:

1. position of the galaxy, in either "chip" or "sky" coordinates
2. a shear to be applied to each galaxy
3. a mask
4. noise properties
5. WCS
6. PSF model

Each of the measured quantities can have associated errors.

Catalogs will be presented at FITS tables.

5.2 PhoSim input catalog

This type is defined by PhoSim.

QUESTION: *Put a brief description of a PhoSim input catalog here. Are they FITS files? What are the data that are contained?*

5.3 Point Spread Function

There are several common representations for a PSF:

1. shapelets
2. gaussian mixtures (sum of gaussians of various sizes and centers)
3. pixel-based (an image of the PSF can be drawn at any location)

QUESTION: *What other representations of PSFs are important?*

PSFs should always have an associated uncertainty. Typically, this uncertainty always represented as a covariance matrix.

5.4 Image

Images are always FITS images. Different types of image include:

Postage stamp image: A postage stamp images includes only one object (*e.g.*, a galaxy).

Color image: ***QUESTION:*** *What is the special feature of a color image? Does this mean a FITS file containing several images of the same part of sky, one for each of several color bands?*

5.5 Mask

A mask provides one number (in the range $0 - 1$) per pixel, that describes what fraction of the light from galaxies in that pixel would be observable.

For a color image, there may be a mask for each color channel.

QUESTION: *For n color channels, does this mean n matrices, or one matrix in which each element is a vector of length n? Or something still different?*

Common mask formats are *mangle* and *HEALPix*.

5.6 World Coordinate System

The chip (x, y coordinates in an observed image) and sky (ra and dec on the sky) are related by the WCS, which is essentially just a pair of functions: ra(x,y) and dec(x,y) (or the inverse).

Bibliography

[1] "The lsst desc task 2." [Online]. Available: https://confluence.slac.stanford.edu/display/LSSTDESC/Task+2

[2] "The LSST desc." [Online]. Available: https://confluence.slac.stanford.edu/display/LSSTDESC/Home

[3] "The lsst desc task 3." [Online]. Available: https://confluence.slac.stanford.edu/display/LSSTDESC/Task+3